SUBJECT: SOFTWARE DESIGN 2	ANNEXURE(S) Formulas	PAGES 3	TIME 3 HRS
CODE: SDN260S	DATE 15 August 2022	MARKS 78	

Cape Peninsula University of Technology

FACULTY OF ENGINEERING

ASSESSMENT I: SEMESTER TWO

COURSE: BET: COMPUTER ENGINEERING

EXAMINER	:	Haltor Mataifa
MODERATOR (INTERNAL)	:	Mr. V. Moyo
MODERATOR (EXTERNAL)	:	N/A

SPECIAL INSTRUCTIONS

1. Answer all questions
2. Write your name and student number on each page of everything you submit
3. Write answers to theoretical questions in an MS Word document
4. Create a new project for each programming problem
5. Zip all files and folders together and submit as one zipped file on Blackboard
6. Be sure to add comments to your program code to make it understandable

QUESTION 1 **[23]**

1.1. Explain the main difference between a String and a StringBuilder. Also state when it would be advisable to use the one instead of the other [**2**]

1.2. Explain the difference between a String literal and a String object instantiated using the new operator in Java [2]

1.3. Explain the difference between string methods equals and compareTo [2]

1.4. Consider the following piece of code:

```
1  String s1=new String("Welcome to SDN260S");
2  String s2=" welcome to SDN260S ";
3. String s3=new String("Welcome to SDN260S");
4. s3==s1;
5. String s4=s2;
6. s4==s2;
```

1.4.1 What is the purpose of the statement in line 4? What is the result after executing this line? [**3**]

1.4.2 What is the purpose of the statement in line 5? What is the result after executing this line? [**3**]

1.4.3 What is the purpose of the statement in line 6? What is the result after executing this line? [**3**]

1.5. What is a *recursive method*? How does it differ from a *standard method*? [2]

1.6. Compare recursion with iteration. What are their similarities and differences? When is it preferable to use recursion, and when should it be avoided? [6]

QUESTION 2: **[20]**

Write a Java application that will request a user to enter a sentence, after which it will perform the following operations on the sentence:

- Firstly, it will split the sentence into individual words; assume that the words of the sentence are separated by white space
- It will then output the words to the screen, each word on a separate line
- The first letter of each word must be capitalized before printing to the screen
- The words will also be written to a text file (each on a separate line) in the order that they are printed on the screen. Be sure that writing subsequent words to the text file does not result in overwriting the previously written words

QUESTION 3: **[20]**

Write a Java application that will validate a telephone number entered by a user. The application should check the validity of the telephone number entered against the following requirements:

i. The format should be (XXX) XXX-XXXX (i.e. ten digits)

ii. The first three digits, enclosed in parentheses, are the area code

iii. The next three digits should be separated from the area code by single white space, and by a hyphen from the last four digits

iv. The area code should always begin with 0

v. The second digit of the area code should lie between 0 and 5

vi. The remaining (eight) digits can lie anywhere between 0 and 9

vii. If the user enters an invalid telephone number, the application should inform the user and grant them two more attempts to enter a telephone number in the correct format

viii. After three unsuccessful attempts, the application should output an appropriate message to the user and terminate the program

ix. Once the user has entered a telephone number in the required format, the application should thank the user and print the entered telephone number to the screen in the following format (on separate lines):

    Area code: XXX

    Phone number XXX-XXXX

x. The application needs to *show the user the correct format in which the telephone number needs to be entered* (although it need not inform the user about the limits on the digits as specified in (iv) – (vi))

QUESTION 4: **[15]**

Write a Java application that will use a recursive method to *add all the odd numbers from 0 up to the* (positive integer) *value entered by the user*. The application will do the following:

- Request the user to enter the (positive integer) number which is the upper limit for the computation. For example, if the user enters number=10, the application will add all odd numbers between 0 and 10
- Use the recursive method for computing the sum of odd numbers
- Output the result to the screen as "sum of odd numbers between 0 and n = result", where n is the number entered by the user, and result is the result obtained by applying the recursive method

End of Assessment